DESARROLLO DE UN PROTOTIPO WEB PARA LA CREACIÓN DE DIAPOSITIVAS BASADAS EN COMPONENTES

JOHN DARWIN MORALES GONZÁLEZ

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS
INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
SEDE PEREIRA, RISARALDA
AÑO 2017

DESARROLLO DE UN PROTOTIPO WEB PARA LA CREACIÓN DE DIAPOSITIVAS BASADAS EN COMPONENTES

JOHN DARWIN MORALES GONZÁLEZ
Código : 1112777672

TRABAJOS DE INVESTIGACIÓN FORMATIVA:
PROYECTO DE APLICACIÓN

ASESOR METODOLÓGICO:
CARLOS ALBERTO OCAMPO SEPULVEDA

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS
INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
SEDE PEREIRA, RISARALDA
AÑO 2017

Nota de aceptación:

_____
_____
_____
_____
_____
_____
_____

_____
Firma del Asesor metodológico

_____
Firma del Jurado

_____
Firma del Jurado

Pereira, Risaralda - 25 de Mayo de 2017

DEDICATORIA

La realización de este trabajo de grado se lo dedico a mi familia que inculcaron grandes valores como el amor al estudio, la honestidad, el respeto y principios espirituales, que me convierten en una persona capaz de aportar positivamente dentro de la sociedad. Igualmente agradezco el esfuerzo de cada uno por apoyar mi permanencia en el alma mater.

También a cada una de las personas que me han acompañado, y brindaron su respeto y cariño en cada uno de los instantes de esta grandiosa carrera.

De la misma forma, a los grandes maestros que establecieron grandes retos y forjaron en mí grandes motivaciones para alcanzar grados de aprendizaje mucho más altos.

CONTENIDO

pág.

LISTA DE TABLAS

pág

LISTA DE FIGURAS

pág

INTRODUCCIÓN

Actualmente las tecnologías de la información y la comunicación (TIC), se vienen desarrollando por el incremento de herramientas informáticas a través de la Web, apoyándose sobre la infraestructura de Internet, dichas herramientas permiten idear, transmitir, crear, manipular o almacenar información de manera remota[1]. Adicionalmente y de manera complementaria también se ha incrementado el uso de dispositivos móviles para fortalecer dichas actividades.

Cada día los sistemas de información se han vuelto más útiles y cotidianos, a medida que se vienen acondicionando para acceder a contenidos, servicios y aplicaciones en la Web. Es así como dentro de estos encontramos las presentaciones multimedia tales como: Power point, Google Slides, Prezzi y Reveal.JS. El primero es considerado como pionero entre las ayudas para la presentación multimedia ejecutada de manera local en un computador personal,[2] dado que su desarrollo comenzó a mediados de los años 80 por la compañía forethougth, para luego ser comprada por Microsoft. Con el tiempo este tipo de herramientas fue evolucionando para ser más accesible sin importar la ubicación geográfica, tal es el caso de herramientas como Google Slides, PowerPoint online y Prezzi, que se adaptaron para ser accesibles por medio de la Web, estando en la categoría de Software como servicio.

Actualmente, gracias al desarrollo de código abierto y algunas herramientas como Github, siendo una plataforma web de desarrollo colaborativo con carácter público para alojar proyectos de forma remota utilizando el sistema de control de versiones Git[3]. Este tipo de herramientas han contribuido a la comunidad de desarrollo y que a su vez también resuelven necesidades informáticas y tecnológicas. Tal es caso de Reveal.JS[4] que es uno de los

[1] Sobre el incremento del uso de las TIC [online]. consultada 16 Sep 2015 http://www.enticconfio.gov.co/index.php/actualidad/254-tendencia-yuso-de-las-tic-en-colombia.html

[2] Sobre la historia de los programas de presentación asistidos por computador.[online] consultada 16 Sep 2015 http://presentacionesenpowerpointsc.blogspot.com.co/2009/09/la-historia-de-powerpoint.html, http://es.ccm.net/faq/9329-powerpoint-historia

[3] Introducción a Git [online].consultada 16 Sep 2015 http://conociendogithub.readthedocs.org/en/latest/data/introduccion/

[4] RevealJS, página oficial del framework, consultada http://lab.hakim.se/reveal-js/#/

proyectos de código abierto que se encuentran alojados en esta plataforma, que permite hacer determinadas tareas para la presentación de diapositivas haciendo uso de tecnologías Web, como javascript, css y html; sin embargo revisando el código de esta aplicación se puede prever que carece de un diseño basado en componentes, ya que de esta manera permitirá ser más evolutivo y flexible.

A consecuencia de lo anterior, se utilizará como base el Framework RevealJS para desarrollar un prototipo web para la creación de diapositivas basadas en componentes, utilizando AngularJS en su versión 1, el cual es uno de los frameworks desarrollados por Google, para el desarrollo web sobre lenguaje javascript.

Además de lo anterior, este documento contiene de acuerdo a las técnicas de presentación y exigencias de la universidad, la estructura formal de la propuesta del anteproyecto de grado, para el inicio del trabajo de formación investigativa, proyecto de aplicación, el cual se divide en siete capítulos que abarcan desde la formulación del problema hasta el cronograma para su ejecución. Para la preparación del trabajo se requirió hacer una revisión literaria, conceptual y legal acerca del tema, y a partir de ella diseñar la estrategia metodológica que conlleve a la ejecución de la práctica.

1. DEFINICIÓN DEL PROBLEMA

1.1 PLANTEAMIENTO

Entendiendo que RevealJS, es un de los Framework más utilizados en la actualidad al permitir a los usuarios con conocimientos básicos en tecnologías como HTML, CSS y Javascript, la creación de diapositivas Web. Su desarrollo es poco escalable para los contribuyentes que quieran crear nuevos componentes e implementar otras funcionalidades que podrían enriquecerlo, de modo que, dichos cambios satisfagan la presentación en tiempo real entre un expositor y la audiencia en forma remota. Esto podría aumentar la experiencia en el proceso de esta comunicación a través de presentaciones con diapositivas en la Web.

1.2 FORMULACIÓN

Con lo anterior descrito es preciso definir ¿De qué manera renovar la creación de diapositivas web para crear componentes e implementar nuevas funcionalidades, con base en el framework RevealJS?

1.3 SISTEMATIZACIÓN

Según la formulación y explorando más en detalle la pregunta general, subdivide una serie de interrogantes tales como: ¿De qué manera agregar nuevas funcionalidades, evitando problemas de fuerte acoplamiento dentro del sistema?, ¿De qué manera estandarizar el desarrollo del lado del cliente? y ¿Cómo conseguir que los componentes emitan eventos en tiempo real?.

2. JUSTIFICACIÓN

Actualmente los recursos tecnológicos y herramientas en la “nube” han creado una revolución en el mundo informático positivamente, donde la ubicación geográfica ya no es tanto una limitante para acceder a la información, más aún cuando esta información está actualizada y siendo guiada por un experto o profesional del tema a tratar. Puesto a esto, existen herramientas como por ejemplo RevealJS para crear diapositivas web, que permiten presentar ideas o información de una manera sencilla a un público local y además da esta una solución a un servicio básico dirigidas a un público remoto. Sin embargo esta herramienta carece de un servicio donde los elementos internos dentro de este puedan tener un rol, aún más interactivo dentro del contenido, donde el expositor pueda manipular y

cambiar el estado de un elemento, que podrá percibir remotamente su público en tiempo real.

Por lo anterior, el trabajo a realizar comprende desarrollar un prototipo Web que responda a las necesidades de tener elementos que emitan eventos en tiempo real, por lo cual se requiere evaluar el estado actual del framework RevealJS, para ser tomado como referencia y desarrollar una estructura basada en componentes y aplicar un patrón de diseño, lo cual mejoraría la abstracción a un nivel más alto del software, ya que este prototipo pretende dejar el código abierto para el desarrollo colaborativo de la comunidad, donde debe existir un lenguaje de abstracción en común entre los desarrolladores, para mejor entendimiento del diseño.

3. OBJETIVOS

3.1 GENERAL

Desarrollar un prototipo web diferente a RevealJS para la creación y presentación de diapositivas basado en componentes, gestionando elementos de contenido, eventos, y estados realizados por un expositor online a la audiencia remota.

3.2 ESPECÍFICOS

- Analizar el estado actual del framework RevealJS.
- Desarrollar una estructura basada en componentes.
- Adecuar un patrón de diseño sobre el lado del cliente.
- Modelar e implementar un diseño para conseguir que los componentes emitan eventos en tiempo real.

4. MARCO DE REFERENCIA

4.1 ANTECEDENTES

Cada día los sistemas de información se han vuelto más útiles y cotidianos, a medida que se vienen acondicionando para acceder a contenidos, servicios y aplicaciones en la Web. Con ello las aplicaciones de escritorio y aplicaciones móviles como los navegadores web se han ido adaptando, desde sus inicios

en los años noventa con la aparición de la web, tal como lo comenta el profesor de la Universidad de Alicante (España), Sergio Luján Mora [5].

La creación y presentación de diapositivas ha venido cambiando debido al auge de recursos tecnológicos, tal es el caso del Framework RevealJS que permite la creación de diapositivas usando HTML, CSS, y Javascript. Este framework ha sido desarrollado desde el junio cinco(5) de 2011[6], tal como aparece en el histórico de contribuciones siendo el usuario Hakim El Hattab, es de Estocolmo, Suecia , con más contribuciones en este proyecto, dedicado al desarrollo del lado del cliente (front end).

Con lo anterior, al consultar estudios de la Universidad Tecnológica de Pereira, se ha evidenciado que no hay información suficiente sobre el desarrollo de aplicaciones que tengan que ver con la creación y presentación de diapositivas, utilizando tecnologías web. Sin embargo vale aclarar que dentro del alma mater existen diferentes información teóricas que puedan contribuir, a la Ingeniería del Software y la construcción de modelos de aplicaciones distribuida.

4.2 MARCO CONTEXTUAL

Actualmente existen espacios virtuales colaborativos para desarrollar software, los cuales tienen contribuyentes de distintas partes del mundo. Estos espacios colaborativos como GITHUB, permiten crear ambientes de trabajos públicos donde priman las ganas por aprender y que hace más enriquecedor e interesante el producto.

Es por estas razones que el proyecto está enfocado a las distintas personas, con conocimientos básicos en el desarrollo de aplicaciones web, que sean amantes de las tecnologías como HTML, CSS y Javascript.

Sin embargo, para limitar el alcance que tiene el proyecto, es enfocado a la comunidad de desarrollo de aplicaciones web, que estén interesados en el aprendizaje y aplicación de modernas prácticas de desarrollo, del lado del Cliente como del Servidor. Donde dicho alcance está orientado a favorecer el aprendizaje académico de los estudiantes de Ingeniería de Sistemas y

[5] Sergio Luján Mora, Curso "Introducción al desarrollo web": el primer navegador web de Tim Berners-Lee, Mosaic, el primer navegador web desarrollado en NCSA.

[6] Hakimel, usuario Github.[repositorio-online] Reveal.JS https://github.com/hakimel/reveal.js.git

Computación de la Universidad Tecnológica de Pereira, ubicada capital del departamento de Risaralda.

4.3 MARCO CONCEPTUAL

4.3.1 Diapositiva

Las diapositivas han sido una de las herramientas preferidas por la sociedad actual para comunicar de manera gráfica y estructurada información. No obstante con la aparición de herramientas de software para realizarlas se ha incrementado más su uso, permitiendo que sectores como la educación se adopten estas prácticas y eleven el enfoque de sus estudiantes sobre un tema en específico. Es allí donde toma sentido dar soporte por medio de nuevas tecnologías, para crear funcionalidades que aporten a la interacción que existe entre el emisor y el receptor del mensaje.

4.3.2 Componente de software

Un componente dentro del contexto del software, es un elemento de un sistema que ofrece un servicio predefinido, y es capaz de comunicarse con otros componentes.

Los componentes, durante el diseño de un producto de software permite tener abstracción elevada de como esta compuesto un sistema, donde se permite ver la interacción dentro dentro de este.

Llevado al desarrollo del producto, permite obtener una encapsulación de una función en particular, que se comportan como piezas sustituibles, reusables dentro de un sistema, logrando con esto una forma más práctica de conseguir un acoplamiento débil en el comportamiento del sistema.

5. METODOLOGÍA

5.1. PRIMERA ETAPA : Análisis de requerimientos

5.1.1. Requerimientos

1. Se requiere crear una aplicación web para crear diapositivas utilizando html, css y javascript.

2. Cada diapositiva debe tener :
   Una plantilla base donde se colocará el contenido

Esta debe ser creada con html, utilizando clases en sus etiquetas las cuales deben estar dentro del archivo css base.

Transición básica:

Una transición básica entre diapositiva.

3. Como primera versión, la edición de una presentación no se tendrá una interfaz gráfica, todo se deberá hacer a partir de código utilizando html y css.

4. La parte funcional de los elementos de las diapositivas serán manejados con javascript, así que no es necesario que se ingrese algún tipo de código para controlar las funcionalidades preestablecidas. Sin embargo se requiere que el software pueda ir incrementando nuevas funcionalidades con mucha facilidad.

5. En la presentación de diapositivas estas deben tener botones para adelantar o ver la vista previa.

6. Solo por el momento se crean los siguientes componentes básicos que emiten eventos en tiempo real, las cuales son percibidas por los usuarios que ven la presentación.

   Listas numeradas:
   Al hacer clic sobre un elemento de la lista, este se subrayará y se mostrará a todos los usuarios que estén viendo la presentación.

   Gráfica circular:
   Al pasar el mouse sobre el punto de la gráfica, se disparará un en evento de mouse sobre todos los usuarios que estén percibiendo la presentación.

   Componentes para tener en cuenta en el futuro:

   Reproductor de Video:
   Tendrá funcionalidades de adelantar, detener video, y reducir volumen del mismo. La reducción de volumen será igualmente percibida por los usuarios que estén conectados a la diapositiva.

   Reproductor de sonido:
   Tendrá funcionalidades de adelantar, detener audio, y reducir volumen del mismo.

   Tabla:
   Esta tabla tendrá una lista de datos ingresados previamente, los cuales pueden ser ordenados por el presentador y a su vez es percibidos por los usuarios. Contará con un input de búsqueda.

7. Para este prototipo solo existirá una sola presentación por emisor, para mostrar las funcionalidades básicas del comportamiento en tiempo real, la cual será cargada como un archivo html.
8. Un usuario que quiera subir una presentación, deberá estar registrado.

9. Para Crear una diapositiva se necesitará descargar la plantilla base de un repositorio en github.

10. Para subir una presentación el código de la presentación se deberá extraer el contenedor html base del contenido principal.
    - Si existen imágenes dentro de la presentación, estas deben estar deben usar un link externo.
    - El contenedor base estará bien definido como una etiqueta html.
    - Para este prototipo no se tendrá algún tipo de compilador para código html.

11. El usuario presentador determinará en qué momento se comienza a transmitir su presentación.

12. Para ingresar a ver alguna presentación, el usuario deberá tener un registro básico nickname y quedará como usuario anónimo.

13. Existirá un panel de presentaciones que se están transmitiendo en vivo.

14. El Usuario podrá escoger qué presentación quiere tomar.

15. Sobre la presentación irán apareciendo todos los usuarios que estén conectados a una diapositiva.

16. La aplicación debido a que es prototipo incremental, se debe tener en cuenta los componentes restantes para seguir aumentando las características ofrecidas.

17. La aplicación tendrá versión libre a la comunidad de desarrollo para posteriores mejoras.

18. Pueden existir muchos componentes, por presentación que estén capacidad de emitir eventos en tiempo real.

19. Los componentes creados se pueden utilizar en diferentes diapositivas de una misma presentación.

5.1.2. Definición de casos de uso

5.1.2.1. Definición de Roles

Presentador: Es el usuario quien crear las presentaciones utilizando HTML y CSS a partir de plantillas, utilizando los componentes proporcionados por la aplicación para mostrar la información en cada diapositiva. Se encarga de subir el contenido de la presentación a la plataforma. Por consecuencia tiene el control de la presentación que está transmitiendo.

Espectador: Usuario que ingresa a las diapositivas proyectada por el presentador, y el cual observa eventos que son emitidos y controlados por el presentador para transmitir su contenido.

5.1.2.2. Descripción de casos de uso

Tabla 1. Caso de uso 1

Caso de uso No 1	
Caso de uso	Crear una presentación
Actores	Presentador
Propósito	Este caso de uso permite al usuario quien quiere hacer emisión de una presentación, crear una presentación utilizando etiquetas html
Resumen	Este caso de uso comienza cuando el usuario ya ha descargado la presentación de repositorio de la plantilla base de la presentación.
Tipo	Esencial
Referencias	Historia de usuario 9
Prerrequisito	Descargar la plantilla base
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1.Este caso de uso inicia cuando el usuario ha descargado la plantilla base del repositorio.	
2. Agrega el contenido para cada diapositiva, utilizando opcionalmente los componentes preestablecidos.	
3. Abre el archivo html base para la presentación.	**4**. Previsualiza que la presentación sobre el navegador.

Curso alterno

Tabla 2. Caso de uso 2

Caso de uso No 2	
Caso de uso	Registrarse como usuario emisor
Actores	Presentador
Propósito	El usuario que quiera emitir su presentación, deberá registrarse con datos básicos.
Resumen	Cuando el usuario se encuentra en la plataforma, y quiera emitir su presentación deberá registrarse.
Tipo	Esencial
Referencias	Historia de usuario 8, 2
Prerrequisito	Ingresar a la plataforma
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1. Este caso de uso inicia cuando el usuario ya ha ingresado sobre la plataforma	
2. Clickea el botón emitir presentación	**3**. Despliega una ventana modal con un registro básico.
4. El usuario llena el formulario.	**5**. Muestra un mensaje que ya ha sido registrado.
	6.Redirecciona a la vista de subir presentación
Curso alterno	
Acción 3: Si el usuario ya se encuentra registrado no muestra la ventana modal y pasa a la acción 6. **Acción 4:** El sistema da una opción a través de un link de loguearse. **Acción 5:** Si el usuario existe muestra un error de existencia del usuario.	

Tabla 3. Caso de uso 3

Caso de uso No 3	
Caso de uso	Subir una presentación
Actores	Presentador
Propósito	El usuario podrá subir su presentación
Resumen	Para subir la presentación, se deberá copiar el contenido base determinado por una etiqueta

	html. Esta debe ser copiada en un campo de texto.
Tipo	Esencial
Referencias	Historia de usuario 10
Prerrequisito	Caso de uso 2.
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1. Este caso de uso inicia cuando el usuario está sobre la vista de subir presentación.	
2. El usuario copia el contenido base dentro junto con la etiqueta: <content-base></content-base>	
3. Luego pega el contenido sobre el text-area de la interfaz, y guarda los cambios.	**4.** Se muestra un mensaje de éxito
5. El usuario da click en la opción ver presentación.	**6.** Se redirecciona a una vista para ver la presentación.
Curso alternos	
Acción 4: Si no fue guardado con éxito se mostrará un mensaje de error.	

Tabla 4. Caso de uso 4

Caso de uso No 4	
Caso de uso	Transmitir presentación
Actores	Presentador
Propósito	El presentador determinará cuándo transmitir presentación
Resumen	El presentador comienza a transmitir esta presentación, será mostrada en un panel general para que los usuarios comienzan a seguirla.
Tipo	Esencial
Referencias	Historia de usuario 11
Prerrequisito	Caso de uso 3
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1 .Este caso de uso inicia cuando el usuario tiene previsualizando la diapositiva.	

2. El usuario le da click sobre comenzar presentación	**3.** La presentación se redirecciona a otra url, y comienza a partir de la diapositiva número 1.
	4. El sistema publica la diapositiva en el panel central de diapositivas en streaming.
Curso alternos	

Tabla 5. Caso de uso 5

Caso de uso No 5	
Caso de uso	Emitir eventos en tiempo real
Actores	Presentador, Espectador
Propósito	Siempre y cuando exista una acción sobre un elemento que emita un evento el tiempo real, este debe ser reportado en el canal donde se está transmitiendo la diapositiva.
Resumen	Este caso de uso comienza cuando el presentador ha logrado comenzar a transmitir la presentación
Tipo	Esencial
Referencias	Historia de usuario 7
Prerrequisito	Historia de usuario 4
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1 . Este caso de uso inicia cuando el presentador ha logrado empezar a transmitir la presentación en tiempo real.	
2.El usuario acciona un evento sobre algún componente definido para emitir un evento en tiempo real.	**3.** El componente escucha este este evento.
	4. Es transmitido hacia un servicio receptor de eventos del lado del cliente
	6. El servicio envía el evento al servidor.
	7. El servidor recibe la señal de un evento y detecta que componente lo emitió y sobre que presentació lo emitió.
	8. Emite ese evento sobre todos lo usuarios receptores, registrados en la sala que están viendo la presentación

9. El espectador, puede percibir el cambio de estado de algún componente dependiendo del tipo de acción recibida.	
Curso alternos	
Acción 3: El sistema muestra un mensaje diciendo que el usuario no existe.	

Tabla 6. Caso de uso 6

Caso de uso No 6	
Caso de uso	Ver una presentaciones emitidas
Actores	Usuario Receptor(Público), Usuario Presentador
Propósito	Existe un panel donde se muestran las presentaciones transmitidas en vivo, las cuales se podrán listar sobre esta vista.
Resumen	Este caso de uso comienza con el usuario ha podido ingresar a la plataforma
Tipo	Esencial
Referencias	Historia de usuario 13
Prerrequisito	
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1. Este caso de uso inicia cuando un usuario ingresar a la plataforma.	**2.** El sistema carga todas las presentaciones transmitidas en el momento
3. El usuario puede ver todas las presentaciones en un panel principal.	
Curso alterno	
Acción 2: Si no existen presentaciones para mostrar, el sistema cargará un mensaje indicando que en el momento no hay ninguna presentación para mostrar.	

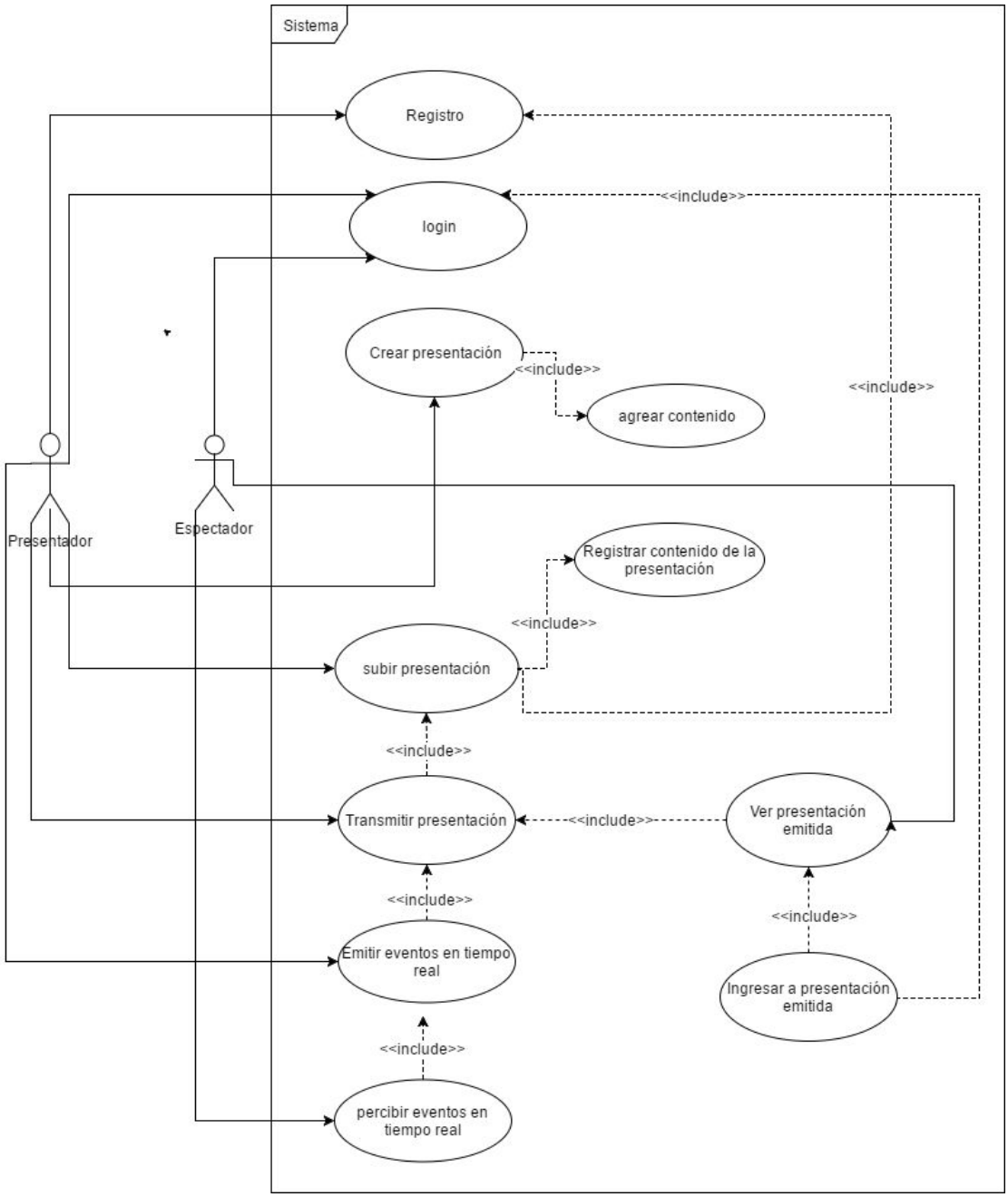
Tabla 7. Caso de uso 7

Caso de uso No 7	
Caso de uso	Ingresar a presentación emitida
Actores	Usuario Receptor(Público)
Propósito	El usuario al cual le interese una presentación que es transmitida en el momento, podrá ingresar una vez se haya registrado con datos básicos, si el usuario ya ha sido registrado podrá revisar la presentación.
Resumen	Este caso de uso comienza cuando un usuario
Tipo	Esencial

Referencias	Historia de usuario 12
Prerrequisito	caso de uso 4
Curso normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1. Este caso de uso inicia cuando un usuario receptor quiera ingresar a ver una presentación en vivo.	
3 . El usuario selecciona una presentación transmitida en vivo.	**4.** El sistema concede el permiso de ingresar a la diapositiva
	5. Carga la presentación en la actual diapositiva, puesta por el presentador.
Curso alterno	
Acción 4: Si el usuario no se encuentra registrado, entonces abrirá una ventana modal de registro básico. **Acción 5:** Si es válido el registro, la ventana modal se cerrará, y redireccionará hacia la URL donde se está transmitiendo la presentación. **Acción 5:** Si no es válido el registro mostrará un mensaje de error de la validación.	

5.1.3. Vistas de casos de uso

Figura 1. Vista casos de uso



5.2 SEGUNDA ETAPA : Diseño y evaluación de la Arquitectura de Software.

5.2.2. Diseño de la arquitectura

5.2.1.1. Atributos de Calidad

Según Mario R. Barbacci, investigador en áreas de arquitectura de software[7], relacionar atributos de calidad con una arquitectura provee una base para la toma de decisiones objetivas sobre acuerdos de diseño.[8]

Es así como Leonard Joel Bass, ingeniero de software miembro Senior del SEI (Software Engineering Institute)[9],establece una clasificación de los atributos de calidad en observables y no observables. Los observables se determinan del comportamiento en tiempo de ejecución. Por otro parte los no observables son los que se determinan durante el desarrollo del sistema.

Por lo tanto, para este caso tomaremos los **no observables**, ya que se pueden precisar en esta etapa de desarrollo del diseño del sistema, y para este caso en una etapa temprana antes de ir al detalle del diseño de la arquitectura.
Dentro de los atributos de calidad no observables, descritos por Bass se encuentra:

- Configurabilidad
- Integrabilidad
- Integridad
- Interoperabilidad
- Modificabilidad
- Mantenibilidad

---

[7] "Mario R. Barbacci • IEEE Computer Society." [online] https://www.computer.org/web/awards/merwin-mario-barbacci. Fecha de acceso 17 abr.. 2017.

[8] Erika Camacho, Fabio Cardeso, Gabriel Nuñez. Guía Arquitectura de Software.[documento digital] Calidad de la arquitectura p. 8.

[9] "About Us | Our People | Staff Profile - Software Engineering Institute." 3 abr.. 2017, http://www.sei.cmu.edu/about/people/profile.cfm?id=bass_13085. Fecha de acceso 17 abr.. 2017.

- Portabilidad
- Reusabilidad
- Escalabilidad
- Capacidad de Prueba

(Véase Guía Arquitectura de Software,.calidad arquitectónica - Tabla 2 pág.10)

Para determinar qué atributos de calidad son los más importantes debido a las necesidades planteadas en los requerimientos de la aplicación, se tomarán los requerimientos no funcionales y se hará una comparación con la definición de los atributos de calidad no observables descritos por Bass

Tabla 8. Relación atributos de calidad con los requerimientos

No	Requerimientos	Atributos de Calidad
16	La aplicación debido a que es un prototipo incremental, se debe tener en cuenta los componentes restantes para seguir aumentando las características ofrecidas.	Mantenibilidad
17	La aplicación tendrá versión libre a la comunidad de desarrollo para posteriores mejoras.	Modificabilidad
18	Pueden existir muchos componentes, por presentación que estén capacidad de emitir eventos en tiempo real.	Escalabilidad
19	Los componentes creados se pueden utilizar en diferentes diapositivas de una misma presentación.	Reusabilidad

Tabla 9. Descripción de atributos de calidad seleccionados

Atributos de Calidad	Descripción
Modificabilidad (Modifiability)	Es la habilidad de realizar cambios futuros al sistema. (Bosch et al. 1999).
Escalabilidad (Scalability)	Es el grado con el que se pueden ampliar el diseño arquitectónico, de datos o procedimental (Pressman, 2002).
Mantenibilidad (Maintainability)	Es la capacidad de someter a un sistema a reparaciones y evolución (Barbacci et al., 1995)
Reusabilidad (Reusability)	Es la capacidad de diseñar un sistema de forma tal que su estructura o parte de sus componentes puedan ser reutilizados en futuras aplicaciones (Bass et al. 1998).

Tabla 2 pág.10 - Guía Arquitectura de Software, 3.calidad arquitectónica

Estos estos atributos de calidad como se mencionó al inicio, se tomarán para guardar una relación con la arquitectura, y así tomar decisiones sobre el diseño de la aplicación.

5.2.1.2. Patrones y estilos arquitectónicos

Los patrones y estilos arquitectónicos juegan un papel importante en el desarrollo del diseño; según Bosch estos pueden mejorar o disminuir las posibilidades de satisfacción de ciertos atributos de calidad. Lo cual se puede interpretar que dependiendo de los atributos de calidad escogidos se puede implementar cierto estilos y patrones utilizados en la arquitectura.

Aunque se debe tener en cuenta que los atributos de calidad dependen de su contexto en el que son especificados y los objetivos para los cuales son establecidos, sin embargo como caso de estudio se establecen unos atributos de calidad de acuerdo a los estilos y patrones de arquitectura.

Es importante tener en cuenta que para obtener un acercamiento más explícito de los atributos de calidad dependiendo del contexto, se plantean técnicas para evaluar el potencial que tiene la arquitectura de alcanzar estos atributos de calidad. Una de las técnicas para evaluar el diseño establecido, es la evaluación por escenarios, la cual más adelante se desarrollará.

Siguiendo lo planteado, se determinará qué estilo, patrón arquitectónico y patrón de diseño se emplea, siguiendo las recomendaciones realizadas por expertos en el tema.

1. **Estilo arquitectónico**

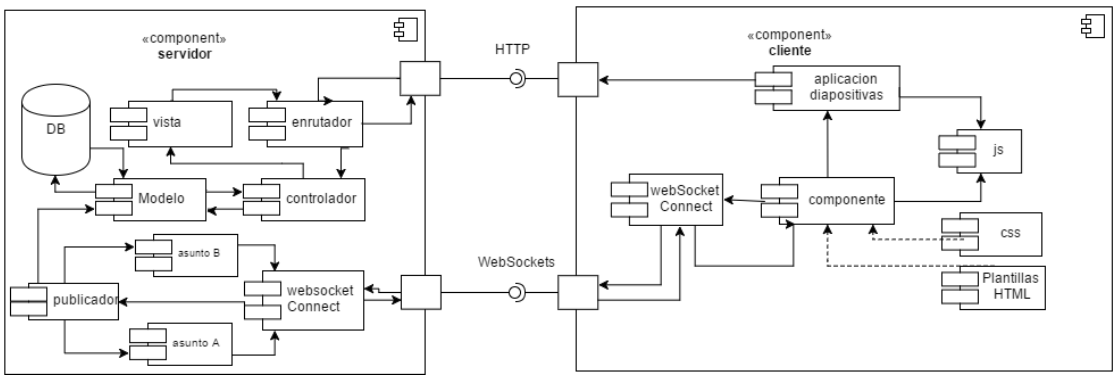
   De acuerdo con los estilos arquitectónicos descritos por Bass en 1998, frente a los atributos de calidad asociados a esta, se puede determinar que el estilo al cual más se acerca, es al de los **componentes independientes**, debido a que está asociado a los atributos de calidad **Modificabilidad** y **Escalabilidad**, los cuales fueron planteados en el punto anterior.[10] Y aparece un tercer atributo relacionado con estos dos atributos y es el de Reusabilidad.

[10] Erika Camacho, Fabio Cardeso, Gabriel Nuñez. Guía Arquitectura de Software [documento digital].Estilo Arquitectónico, Tabla 8.Estilos Arquitectónicos y atributos de Calidad, p.20

Según Bass, el estilo arquitectónico componentes Independientes, consisten en número de procesos u objetos independientes que se comunican a través de mensajes. De igual manera señala que los atributos de calidad con los cuales tiene conflictos son los de desempeño e integrabilidad, los cuales en esta arquitectura no son tomados en cuenta.

La descripción de este estilo arquitectónico, según el contexto de esta aplicación va muy estrecho con la solución de poder integrar componentes que se construyan, definiendo una interfaz de comunicación con su entorno, y así se puedan crear nuevos componentes que sean independientes de donde se utilicen, pero que tengan descrito su forma de comunicación dentro de la aplicación.

Figura 2. Diagrama UML Estilo arquitectónico



2. Patrón arquitectónico

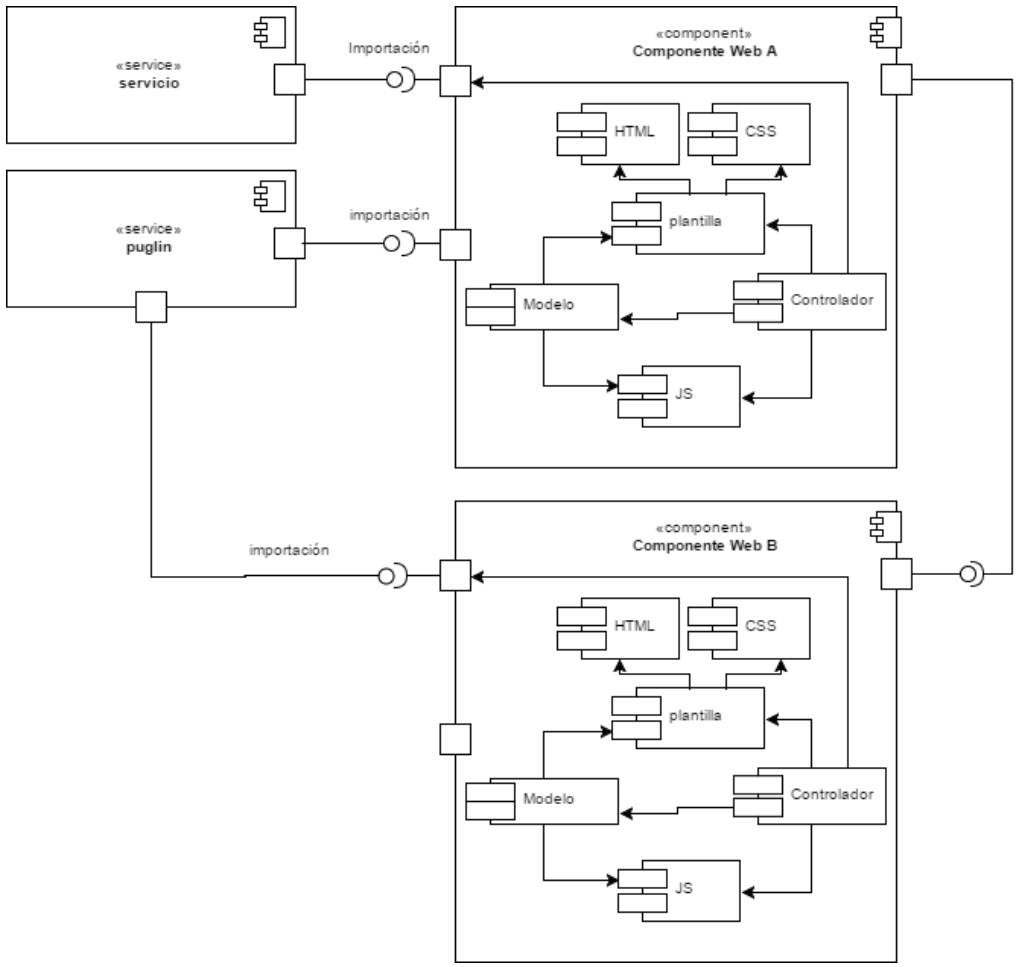
Frank Buschmann, autor del libro Pattern-Oriented Software Architecture, A System of Patterns, en cual propone que los patrones arquitectónicos describen un problema particular y recurrente de diseño, que aparece en contexto de diseño específicos, y presenta un esquema genérico demostrado con éxito para su solución.

Los patrones según esta descripción Buschmann propone que tiene un contexto, en el cual aparece un problema de diseño, el problema es lo que se identifica que aparece de manera recurrente en ese contexto y por último está la solución, que se define como descripción estructural de los componentes que se colaboran entre sí, para mostrar de qué forma se puede afrontar ese problema recurrente.

De esta manera plantea Buschman que cada patrón arquitectónico también tiene una relación estrecha con los atributos de calidad de una arquitectura.

Dentro de los patrones arquitectónicos que más se ajustan a las necesidades y atributos de calidad, es el Modelo Vista Controlador. Ya que trata dos atributos de calidad, que son funcionabilidad y mantenibilidad, juegan un papel importante en la forma que operan los componente web que se pretenden construir.
Este patrón se divide en tres componentes. El modelo (model) contiene la información central y los datos. Las vistas (view) despliegan información al usuario. Los controladores (controllers) capturan la entrada al usuario. Las vistas y los controladores constituyen la interfaz del usuario.

Figura 3. Diagrama UML patrón arquitectónico



3. **Patrón de diseño**

   Los patrones de diseño según Buschmann, buscan describir una estructura comúnmente recurrente en la comunicación entre componentes según un contexto dado.

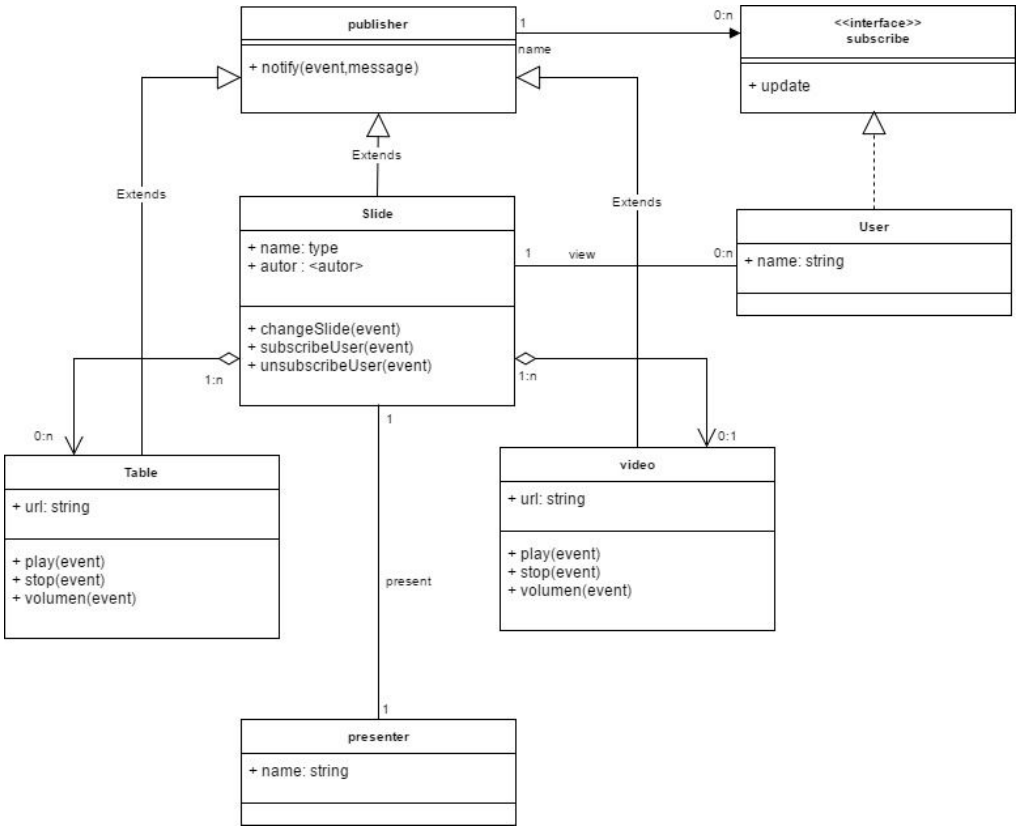
   Es así que para este caso se hace necesario de un patrón de diseño, para la comunicación que tendrán los componentes que emitan eventos en tiempo. Y según los requerimientos es determinante que una diapositiva tenga muchos componentes que emitan eventos en tiempo real, por lo cual se necesita una estructura que vaya de acuerdo a la **escalabilidad**.

   Siguiendo las recomendaciones de Buschman, se utilizará el patrón **publisher/subscriber**, el cual la escalabilidad es un factor que mejora este tipo de diseño.

   El patrón de diseño ayuda a que la comunicación se haga de manera sincronizada. Donde hay una propagación de eventos realizados por un editor (publisher) el cual notifica a un suscriptor (subscriber) sobre los cambios de estado sobre un tema especificado.[11]

Figura 4. Diagrama UML patrón de diseño

[11] Erika Camacho, Fabio Cardeso, Gabriel Nuñez. Guía Arquitectura de Software [documento digital].Patrón de diseño, Tabla 12.Patrones de diseño y atributos de calidad., p.24



5.2.2. Evaluación de la arquitectura

Con base a Jan Bosch experto consultor en la arquitectura de software, determina que la tarea de evaluar una arquitectura es algo que cobra relevancia, puesto que pretende medir propiedades del sistema en base a especificaciones abstractas, y que además los objetivos de la evaluación dependen la situación en la que se encuentre el arquitecto y las técnicas que emplea.

Se plantea entonces el método ATAM (Architecture Trade-off Analysis Method), este revela la forma en que la arquitectura especificada satisface ciertos atributos de calidad, y provee una visión de cómo los atributos de calidad interactúan con otros.[12]

[12] Erika Camacho, Fabio Cardeso, Gabriel Nuñez. Guía Arquitectura de Software [documento digital].Architecture Trade-off Analysis Method (ATAM), p.45-46

El **método ATAM** consiste en 9 pasos divididos en 4 etapas de la las cuales se desarrollaron de la siguiente manera:

FASE 1:

1. Presentación del ATAM :

El método de evaluación ATAM, es una forma de determinar los riesgos potenciales que tiene el momento de diseñar y adoptar cierta arquitectura, de acuerdo con los atributos de calidad propuestos para esta.

Las expectativas que se tienen con esta evaluación temprana es corregir errores antes de que pasen a una fase de codificación de la aplicación y así contrarrestar posibles elevados costos por atrasos en cronogramas.

2. Presentación de las metas del negocio:

En esta evaluación se pretende que los requerimientos no funcionales del sistema, se cumplan, los cuales están condicionados con atributos de calidad previamente descritos, véase *tabla 8. Relación atributos de calidad con los requerimientos*

3. Presentación de la arquitectura

Este proyecto pretende que sea desarrollado basado en componentes, de esta manera esta cada componente representa una parte del sistema que encapsula una función particular, que se compartan como piezas sustituibles y reutilizables dentro del sistema, logrando que estos componentes tengan un acoplamiento débil dentro del sistema.

Para llegar a obtener esta forma de desarrollo se plantea un estilo arquitectónico de componentes independientes, donde se tendrá que definir interfaces para la comunicación entre estos componentes creados.

Igualmente se aplicará el patrones arquitectónico Modelo Vista Controlador, para conseguir que cada componente creado dentro de la aplicación se comporte de una manera más encapsulada.

[13] Erika Camacho, Fabio Cardeso, Gabriel Nuñez. Guía Arquitectura de Software [documento digital].Architecture Trade-off Analysis Method (ATAM),Tabla 18. Pasos del método de evaluación ATAM p.45-46

En esta aplicación además los componentes que serán creados dentro de las diapositivas, se definirán para que emitan eventos preestablecidos, los cuales son transmitidos en tiempo real hacia los espectadores que estén viendo la presentación. Por lo tanto se necesitará de un patrón de diseño que apoye esta forma de comunicación entre los diferentes componentes, por lo cual se determinó que el patrón de diseño sería publisher/subscriber.

Para efecto de profundizar más sobre la arquitectura propuesta, dirigirse a la sección 5.2.1.2 Patrones y estilos Arquitectónicos de esta documento.

FASE 2:

4. Identificación de los enfoques arquitectónicos:

Previamente se ha hecho un análisis donde se describe la los factores de calidad para determinar los enfoques y estilos y patrones escogidos, los cuales son tres:

- Componentes Independientes:
  En este diseño inicial, se describe de manera general como los componentes de la aplicación interactúan entre sí. Sin embargo la esencia del proyecto reside en descomponer y abstraer funcionalidades que permitan ser vistas como un componente, por lo cual más adelante se pretende poder llegar a algo más completo y no tan generalizado.

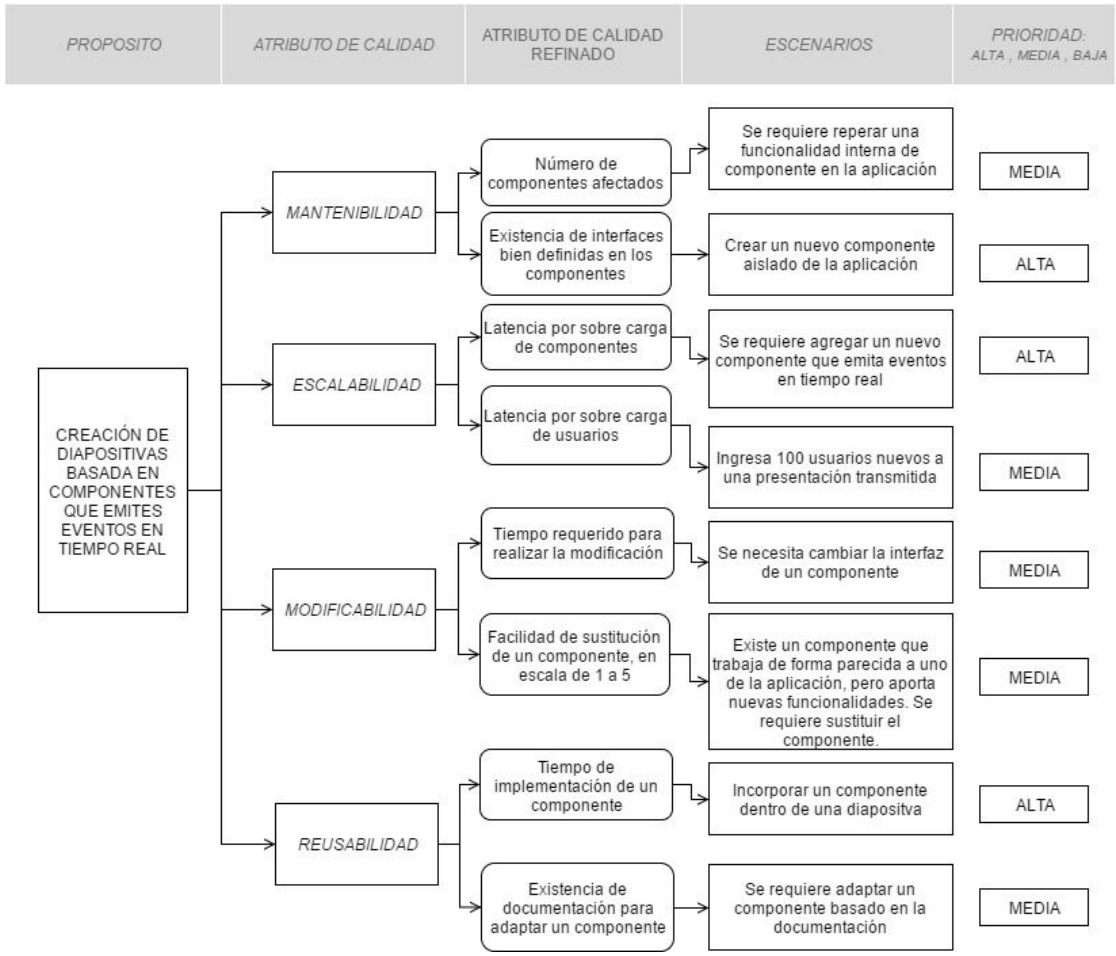
- Modelo Vista Controlador
  Este patrón arquitectónico se adapta mejor a los componentes que requiera interfaz de usuario, ya que se adapta al contexto y la solución de los problemas recurrentes que este patrón brinda. Sin embargo a los componentes que no requieran de una interfaz se podría dejar solo como Modelo Controlador.

- Publisher/Subscriber
  Este patrón de diseño será ampliamente utilizado para todos los componentes que se determinen que emitiran eventos en tiempo real, ya que este patrón resuelve un problema de comunicación sincrona, donde la propagación de eventos en tiempo real juega un papel importante sobre la presentación de diapositivas.

5. Generación del Utility Tree

Se generó el árbol de Utilidad con base a los atributos de calidad obtenidos según los requerimientos de calidad no funcionales propuestos, para luego ser refinados y convertidos en convertidos en escenarios. Estos escenarios ponen a prueba las propuestas arquitectónicas, para comprobar si estas propuestas alcanzan los requerimientos de calidad especificados.

Figura 5. Utility tree (árbol de utilidad)



Los escenarios fueron priorizados según la importancia para alcanzar al éxito del sistema. Se tomó una escala Baja, Media y Alta, luego se escogieron los escenarios más representativos.

Tabla 10. Escenarios priorizados

Priorización	Escenarios	Atributo de Calidad
1	Crear un nuevo componente aislado de la aplicación	Mantenibilidad
2	Se requiere agregar un nuevo componente que emita eventos en tiempo real	Escalabilidad
3	Incorporar un componente dentro de una diapositiva	Reusabilidad
4	Se requiere reparar una funcionalidad interna de componente en la aplicación	Mantenibilidad
5	Se necesita cambiar la interfaz de un componente	Modificabilidad

Para cada escenario se definió un estímulo, contexto y una respuesta. Esto permite entender y definir el significado del atributo de calidad dentro de la arquitectura.

Tabla 11. Escenario 1

Escenario: 1	
Escenario Crudo:	Crear un nuevo componente aislado de la aplicación
Objetivos del negocio correspondientes:	Mantenimiento
Atributo de calidad relevantes:	Mantenibilidad
Estímulo:	Crear un nuevo componente
Fuente de estímulo:	Desarrollador
Entorno:	Aplicación del lado del cliente
Artefacto (si se conoce)	componente emite eventos en tiempo real
Respuesta:	El diseño propone interfaces bien definidas en la aplicación para reducir la cantidad de dependencias entre componentes
Medida de la respuesta:	Existencia de interfaces bien definidas en los componentes
Preguntas:	Máximo número de dependencias
Problemas:	

Tabla 12. Escenario 2

Escenario: 2	
Escenario Crudo:	Se requiere agregar un nuevo componente que emita eventos en tiempo real
Objetivos del negocio correspondientes:	Escalabilidad
Atributo de calidad relevantes:	Escalabilidad
Estímulo:	Agregar un nuevo componente
Fuente de estímulo:	Presentador
Entorno:	Diapositiva
Artefacto (si se conoce)	componente emite eventos en tiempo real

Respuesta:	Se propone un diseño capaz de incrementar la cantidad de componentes que emitan eventos en tiempo y poder administrar esta comunicación
Medida de la respuesta:	Latencia por sobre carga de componentes en una presentación
Preguntas:	Máximo número de componentes permitidos por presentación
Problemas:	

Tabla 13. Escenario 3

Escenario: 3	
Escenario Crudo:	Incorporar un componente dentro de una diapositiva
Objetivos del negocio correspondientes:	Reusabilidad (Reusability)
Atributo de calidad relevantes:	Reusabilidad (Reusability)
Estímulo:	Incorporar un componente
Fuente de estímulo:	Presentador
Entorno:	Aplicación del lado del cliente
Artefacto (si se conoce)	componente
Respuesta:	El tiempo debe ser muy mínimo, ya que se trata de componentes pre-establecidos con parámetros muy definidos
Medida de la respuesta:	Tiempo de implementación de un componente
Preguntas:	
Problemas:	

Tabla 14. Escenario 4

Escenario: 4	
Escenario Crudo:	Se requiere reparar una funcionalidad interna de componente en la aplicación
Objetivos del negocio correspondientes:	Mantenimiento
Atributo de calidad relevantes:	Mantenibilidad
Estímulo:	Reparar componente

Fuente de estímulo:	Usuario encontró un componente con un problema
Entorno:	Aplicación del lado del cliente
Artefacto (si se conoce)	componente emite eventos en tiempo real
Respuesta:	El diseño pretende tener muy separados las funcionalidades por lo cual no hay componentes afectados por la reparación interna.
Medida de la respuesta:	Número de componentes afectados por proceso de reparación de ese componente.
Preguntas:	Número máximo de componentes afectados
Problemas:	

Tabla 15. Escenario 5

Escenario: 3	
Escenario Crudo:	Se necesita cambiar la interfaz de un componente
Objetivos del negocio correspondientes:	Modificabilidad
Atributo de calidad relevantes:	Modificabilidad
Estímulo:	cambio de interfaz
Fuente de estímulo:	Desarrollador
Entorno:	Aplicación del lado del cliente
Artefacto (si se conoce)	Componente
Respuesta:	El diseño arquitectónico propone dividir las vistas de cada componente, lo cual tomaría muy poco tiempo
Medida de la respuesta:	Tiempo requerido para realizar la modificación
Preguntas:	Tiempo promedio para realizar una cambio
Problemas:	

6. Análisis de los enfoques arquitectónicos

Una vez obtenido los escenarios, estos se analizan según los enfoques arquitectónicos, para evaluar la obtención o no del atributo de calidad

requerido, identificando los riesgos, no riesgos, puntos de sensibilidad de los cuales se puede ver afectado la arquitectura y compensación (Tradeoff) para cada caso.

Tabla 16. Análisis propuesta arquitectónica - escenario 1

Análisis propuesta Arquitectónica				
Escenario # :1	Crear un nuevo componente aislado de la aplicación			
Atributo de calidad:	Mantenibilidad			
Decisión arquitectónica	Sensibilidad	Compensación (Tradeoff)	Riesgos	No Riesgos
Cada componente se comunica por medio de una interfaz definida	S1	T1	R1	
Integrar nuevos componentes al sistema	S2	T2	R2	
Razonamiento	S1	Aumenta la capacidad de independencia		
	S2	Se ve un poco afectada la integración de componentes que no tengas las mismas interfaces		
	T1	Aumenta la escalabilidad, pero afecta el desempeño		
	T2	Capacidad de independencia, pero afecta la integrabilidad		
	R1	Que un componente es desarrollado sin tener en cuenta las interfaces		
	R2	Que el sistema falle por el nuevo componente desarrollado		

Tabla 17. Análisis propuesta arquitectónica - escenario 2

Análisis propuesta Arquitectónica				
Escenario #: 2	Se requiere agregar un nuevo componente que emita eventos en tiempo real			
Atributo de calidad:	Escalabilidad			
Decisión arquitectónica	Sensibilidad	Compensación (Tradeoff)	Riesgos	No Riesgos

Los eventos en tiempo real se propagan según el tema al cual un componente esté suscrito.	S1	T1	R1	NR1
Razonamiento	S1	Aumenta la escalabilidad		
	T1	Aumenta la escalabilidad, pero afecta el desempeño		
	R1	No administrar la cantidad de componente puede llevar a bajar el desempeño para la transmisión de eventos		
	NR1	No existe el riesgo que un componente no pueda escuchar mas de un evento		

Tabla 18. Análisis propuesta arquitectónica - escenario 3

Análisis propuesta Arquitectónica	
Escenario #: 3	Incorporar un componente dentro de una diapositiva
Atributo de calidad:	Reusabilidad

Decisión arquitectónica	Sensibilidad	Compensación (Tradeoff)	Riesgos	No Riesgos
Los componentes creados tendrán una interfaz bien definida para ser adaptados a cualquier diapositiva de una presentación	S1	T1	R1	
Razonamiento	S1	Aumenta la posibilidad de tener varios componentes dentro de las diapositivas		
	T1	Es muy independiente dentro de la diapositiva, pero depende de las interfaces del entorno donde será ejecutado		
	R1	No sean declaradas los parámetros que se requieren para el funcionamiento del componente		

Tabla 19. Análisis propuesta arquitectónica - escenario 4

Análisis propuesta Arquitectónica

Escenario #: 4	Se requiere reparar una funcionalidad interna de componente en la aplicación
Atributo de calidad:	Mantenibilidad

Decisión arquitectónica	Sensibilidad	Compensación (Tradeoff)	Riesgos	No Riesgos
El diseño propone que las funcionalidades sean abstraídas en componentes	S1	T1		NR1
Razonamiento	S1	Capacidad de desacoplamiento entre componentes		
	T1	Independencia entre componentes, pero se debe respetar las interfaces de comunicación		
	NR1	No hay riesgos de afectación a otros componentes		

Tabla 20. Análisis propuesta arquitectónica - escenario 5

Análisis propuesta Arquitectónica	
Escenario #: 5	Se necesita cambiar la interfaz de un componente
Atributo de calidad:	Modificabilidad

Decisión arquitectónica	Sensibilidad	Compensación (Tradeoff)	Riesgos	No Riesgos
Cada componente tendrá una vista un modelo y un controlador	S1	T1	R1	
Se propone separar las vistas del modelo de datos del componente y el controlador	S2	T2	R2	
Razonamiento	S1	Aumenta la capacidad de independencia		
	S2	Aumenta la facilidad de mantenibilidad		
	T1	El componente puede ser muy independiente, pero se puede ver afectado estéticamente la interfaz de la aplicación		
	T2	Aumenta la independencia, pero se puede ver afectada la comunicación entre las diferentes partes		

	R1	Afectación estética a la interfaz donde se aplique
	R2	Desempeño en la comunicación por la separación de las partes

FASE 3:

7. Lluvia de ideas y establecimiento de prioridad de escenarios

Este paso requiere del análisis y la lluvia de ideas de los stakeholders del proyecto, y por lo cual no se tiene participación alguna para este proyecto.

8. Análisis de los enfoques arquitectónicos

Este paso requiere repasar los puntos 6 y 7, para confirmar el análisis realizado hasta el momentos, pero al no estar los stakeholders presentes en este proyecto, de igual manera se omite.

FASE 4:

9. Presentación de los resultados

El diseño acerca de la arquitectura hasta el momento se encuentra en desarrollo, por lo cual no existe demasiada información, sin embargo esta evaluación plantea que se puede realizar en una fase temprana de desarrollo, después del diseño básico y justo antes del diseño detallado.

Para los siguientes pasos de este proyecto se propone detallar un poco más la arquitectura, para que en una evaluación futura se pueda tener una mayor profundidad en el desarrollo de esta evaluación.

Siguiendo con la evaluación se encontró que los escenarios planteados describen muy bien casos donde se ponen a prueba el diseño arquitectónico planteado.

Al priorizar los escenarios, uno de los atributo más importante dentro de la aplicación es la mantenibilidad, ya que este atributo está definido en el contexto de creación de componentes con interfaces muy bien definidas, para tener una comunicación desacoplada con otros componentes dentro de la aplicación. Según la evaluación este atributo de calidad se ve en riesgo cuando un nuevo componente es ingresado y no cumple con las interfaces definidas dentro de la arquitectura.

En el segundo escenario, se plantea que se necesita agregar un nuevo componente y que a su vez sea desarrollado para emitir eventos a otros componentes en tiempo real. Se le da una solución arquitectónica, utilizando la propagación de los eventos según el tema al cual un componente esté suscrito, esto cumple con el patrón publisher/subscriber por lo cual cumple con una alta escalabilidad, pero puede verse afectado por el desempeño si no se administra la cantidad de componentes que pueden estar dentro de la aplicación.

Ya en el tercer escenario, se requiere incorporar un componente ya existente a una diapositiva, lo cual está relacionado con la reusabilidad. Dentro de las decisiones arquitectónicas para satisfacer este caso, se requiere que los componentes tengan interfaces bien definidas para ser adaptados a cualquier diapositiva. Un riesgo de realizar esta decisión, es que el usuario no declare los parámetros obligatorios para que el componente trabaje.

En relación con el cuarto escenario, se plantea reparar la funcionalidad interna de un componente, lo cual afecta la mantenibilidad del sistema y por ello la decisión de abstraer funcionalidades importantes del sistema en componentes.

Finalmente en el quinto escenario, se plantea que existen componentes con interfaz de usuario pero que esta interfaz de usuario puede cambiar. Por lo anterior el diseño arquitectónico reconoce que debe existir la separación de una vista modelo y un controlador por cada componente, para que así sea más fácil de modificar dicha interfaz sin afectar la parte funcional del componente en cuestión.
Atendiendo a estas consideraciones, se puede interpretar que la arquitectura así como tiene sus grandes ventajas puede tener riesgos que toca asumir, pero también se pueden reforzar en posteriores diseños de la arquitectura.

5.3 TERCERA ETAPA : Identificación y abstracción de funcionalidades según los requerimientos.

5.3.1 Análisis del estado actual del Framework Reveal JS.

RevealJS es un framework creado aproximadamente en el 2012, por el usuario @hakimel en github, para facilitar la creación de presentaciones usando HTML.[14] El framework se encuentra actualmente en la versión 3.4.1, y está alojado como repositorio en github y según su licencia es totalmente libre, sin embargo se debe cumplir con dos restricciones que más adelante serán mencionadas.

La forma de utilizar el framework, es adjuntado a un documento HTML base un par de estilos css que provee el framework y un archivo principal que es el script donde se encuentra toda su funcionalidad. Para su utilización es necesario leer la documentación que se encuentra muy bien descrita en el repositorio alojado en github.

De forma técnica y analizando el código, una de las características más destacadas de RevealJS, es que este no se basa en ningún script de terceros para trabajar, pero se incluyen algunas librerías opcionales de forma predeterminada. Lo cual le da una independencia bastante importante para cambiar de manera sustancial el código y tener un control sobre la aplicación. Sin embargo al no utilizar algunas librerías base, las cuales pueden facilitar el desarrollo de la aplicación, la estructuración de los datos y el manejo de un estándar para estructurar el código, y por consiguiente facilita la forma de entender la aplicación para nuevos desarrolladores que estén interesados en contribuir con el proyecto.

Por otra parte el framework, es usado del lado cliente, por lo tanto usa javascript que es empleado por defecto por todos los navegadores web, y es un lenguaje multiparadigma. Por lo cual este en su gran parte fue desarrollado bajo el paradigma imperativo, que es caracterizado por manejar una modularización por medio de sus funciones, y estructuración de datos. No obstante una de las desventajas de este paradigma, es la capacidad de crear una abstracción de objetos del mundo real.

Siguiendo con el análisis del código, este está construido aún bajo el estándar Ecmascript 5 - ES5 e , el cual fue lanzado en el año 2011. Aunque no es el más reciente, esto representa una gran ventaja porque las aplicaciones desarrolladas bajo este estándar tienen una compatibilidad de forma nativa del 97,96% en todos los navegadores modernos, y por lo tanto no tendrá conflictos para correr en distintos navegadores. Sin embargo la versión actual

[14] Hakimel, usuario Github.[repositorio-online] Reveal.JS https://github.com/hakimel/reveal.js.git

del estándar Ecmascript 6 - ES6, lanzada en el año 2015, aunque no cuenta con la mayor compatibilidad como el estándar anterior, existen formas de añadir herramientas de desarrollo que soportan esta compatibilidad en los diferentes navegadores.

La importancia del estándar ES6, son las nuevas características que brinda ya que este es mucho más descriptivo a en el momento de aplicar el paradigma orientado a objetos, y mejorar la forma de escribir el código sobre la aplicación, solucionando con mayor facilidad la abstracción de objetos del mundo real.

Otra característica a denotar es que este framework, no usa patrones arquitectónicos, que da solución a problemas recurrentes.

En conclusión con el análisis técnico sobre el desarrollo del framework, así como existen características a favor también hay características que no son tan favorables y que se pueden mejorar. Es por esto que se plantea crear un prototipo diferente al de RevealJS para crear diapositivas, incluyendo además componentes que se puedan emitir eventos en tiempo real, ya que este framework no cuenta con ello. Igualmente se pretende que el prototipo se cree basado en la evaluación arquitectónica realizada en los anteriores puntos.

5.3.2 Abstracción de funcionalidades del Framework Reveal JS

Dentro de este trabajo realizado, como el desarrollo está enfocado basado en componentes, la idea es reutilizar código ya existente, y es por esto que se pretende extraer del framework base partes que puedan aportar al desarrollo de la aplicación.

Debido a lo anterior, se reconocer que en la documentación del framework RevealJS, cualquier persona puede obtener una copia para ser usado, modificado, distribuido, fusionado, publicado, sublicenciado o vendido, sin embargo debe cumplir con dos restricciones:

- Se deberá copiar todo el aviso de copyright en todas las partes del software
- El software se proporciona "tal como está", sin garantía de ningún tipo, expresa o implícita, incluyendo pero no limitado a las garantías de comerciabilidad, adecuación para un propósito particular y no infracción. en ningún caso, los autores o titulares de derechos de autor serán responsables de cualquier reclamación, daño o cualquier otra responsabilidad, ya sea en una acción de contrato, agravio o de otra

manera, que surja de o en relación con el software o el uso o otros negocios en el software.

Por lo anterior se cumplirá con lo expresado en documento de licencia y se aplicará en todas las partes donde este es utilizado.

Partiendo de allí se estuvo revisando qué funcionalidades se pueden extraer, y se determina extraer los siguientes elementos:

- La estructuración de la listas, ya que será uno de los componentes a crear sobre esta nueva aplicación, por lo cual es importante poder extraer lo que se pueda de ello.

- Estilo y reglas de layout para el contenido de los slides, el cual permite tener una distribución de los slides de manera responsive.

5.4 CUARTA ETAPA : Diseño detallado

Con anterioridad se definieron patrones y estilos arquitectónicos, los cuales fueron definidos en una etapa temprana para dar una visión somera del sistema y ser evaluados. Ya para esta etapa se toma como base estos enfoques y se define a nivel más detallado el comportamiento y estructuración del sistema.

Para efectos de entender la estructuración y comportamiento del sistema, se define todas las clases en español, sin embargo en el momento de ser desarrollado se recomienda utilizar nombres en inglés.

5.4.1 Diagrama de clases de la aplicación

Para este diagrama de clases se propone describir de una forma más detallada atributos, métodos y relaciones de cada parte esencial del sistema. Para este caso se puede percibir el patrón de diseño publisher suscribe inmerso, donde la clase Room es la clase principal, y es creada cuando el presentador comienza a transmitir la presentación.

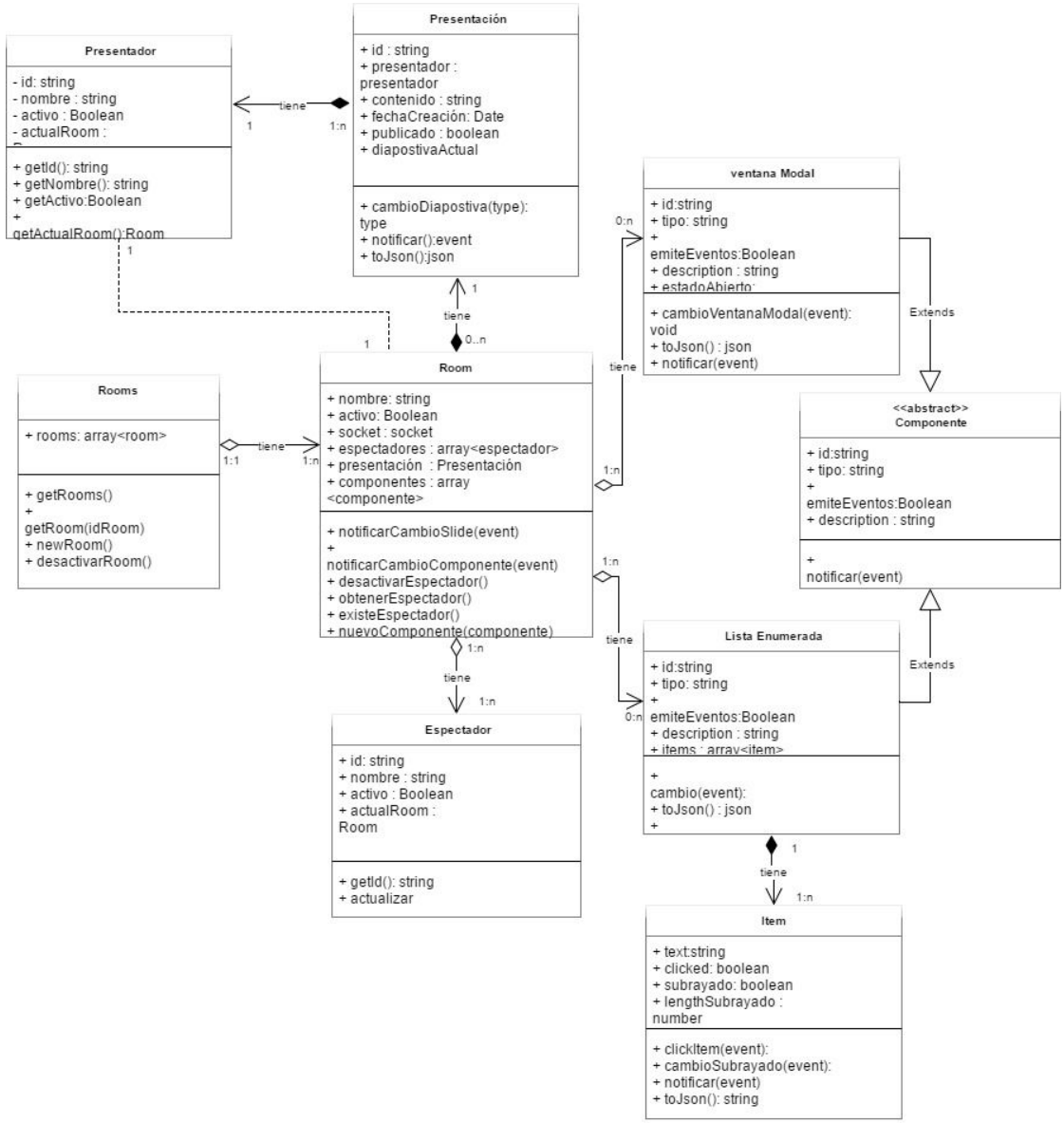
La clase Room, es la encargada de contener la colección de componentes, espectadores y la presentación proyectada. Esta por su característica se encarga de publicar los cambios de los componentes y los cambios de estado de la presentación. Igualmente también se encarga de suscribir y de notificar a los espectadores los diferentes cambios que está detecta en un momento

determinado, ya sea sobre un componente o sobre el cambio del paso de una diapositiva a otra.

La clase Rooms, es la encargada de contener todas las presentaciones que en su momento están siendo publicadas. Esta clase se utilizará para hacer públicas todas las presentaciones sobre el panel de diapositivas.

Los componentes creados, necesitarán ser heredados de la clase abstracta llamada componente, ya que cada componente que emita eventos en tiempo real tiene algunos atributos similares y métodos que ejecutan funciones de manera distinta.

Figura 6. Diagrama de clases de la aplicación



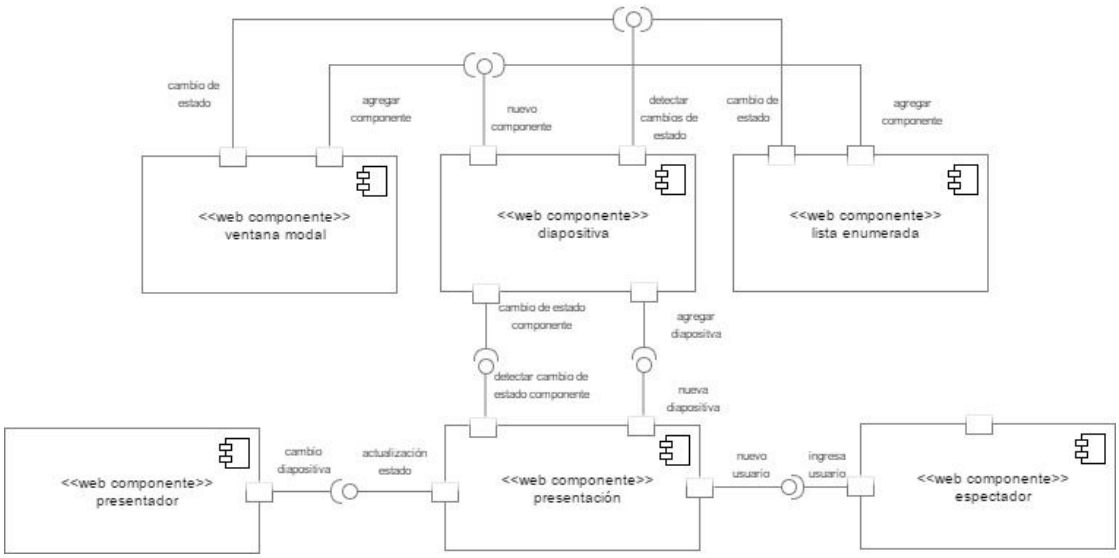
5.4.2 Diagrama de componentes sobre el lado del cliente

Sobre este diagrama de componentes se pretende describir la relación existente entre los diferentes componentes creados del lado del cliente. Estos componentes se describen como web componentes, ya que estos contendrán una interfaz de usuario propia definida sobre una hoja de estilos css, una plantilla html independiente y además será controlado a través de javascript.

La forma de comunicación entre componentes, son por medio de cambios que están siendo observados en el momento de su implementación.

Igualmente este diagrama detalla el estilo arquitectónico, de componente independientes que es utilizado del lado del cliente en la aplicación.

Figura 7. Diagrama de componentes sobre el lado del cliente



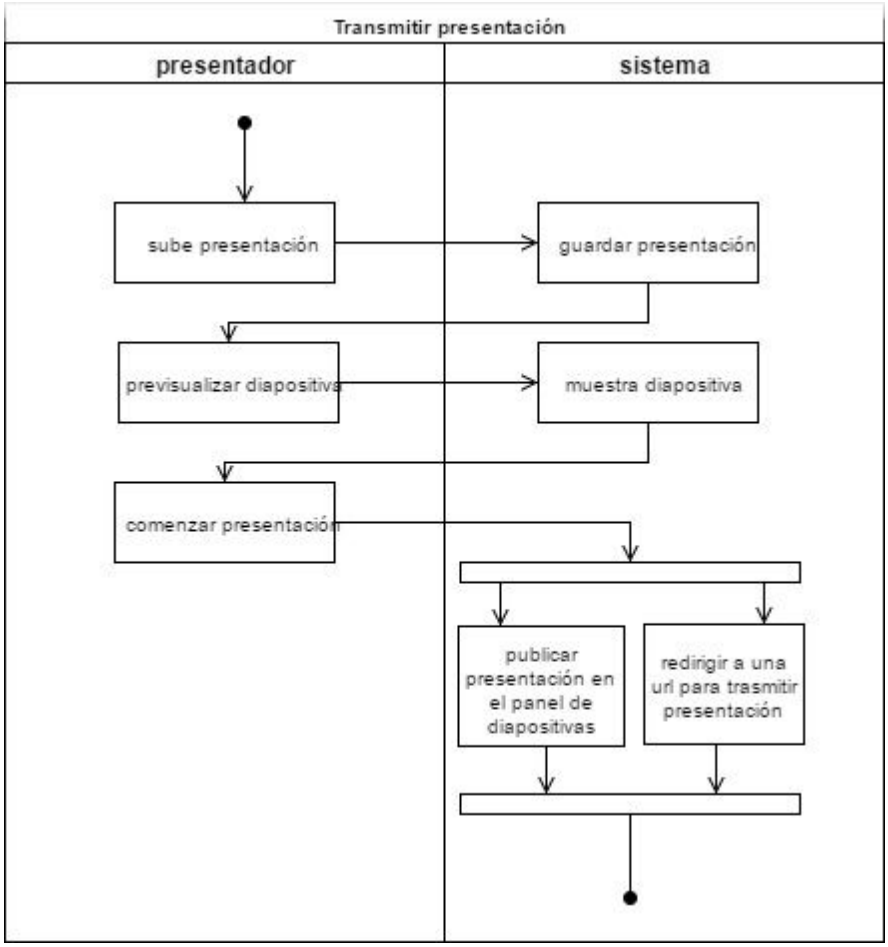
5.4.3 Diagrama de actividades

En este diagrama de actividades se quiere representar dos momentos importante dentro de aplicación, que es la transmisión de las diapositivas y la emisión de eventos en tiempo real de un componente.

5.4.3.1 Transmisión de diapositivas

Esta parte de la transmisión de diapositivas comienza cuando el usuario presentador ya ha creado la diapositiva utilizando html, y a partir de allí se pretende que el usuario suba esta presentación a la plataforma para luego ser transmitida.

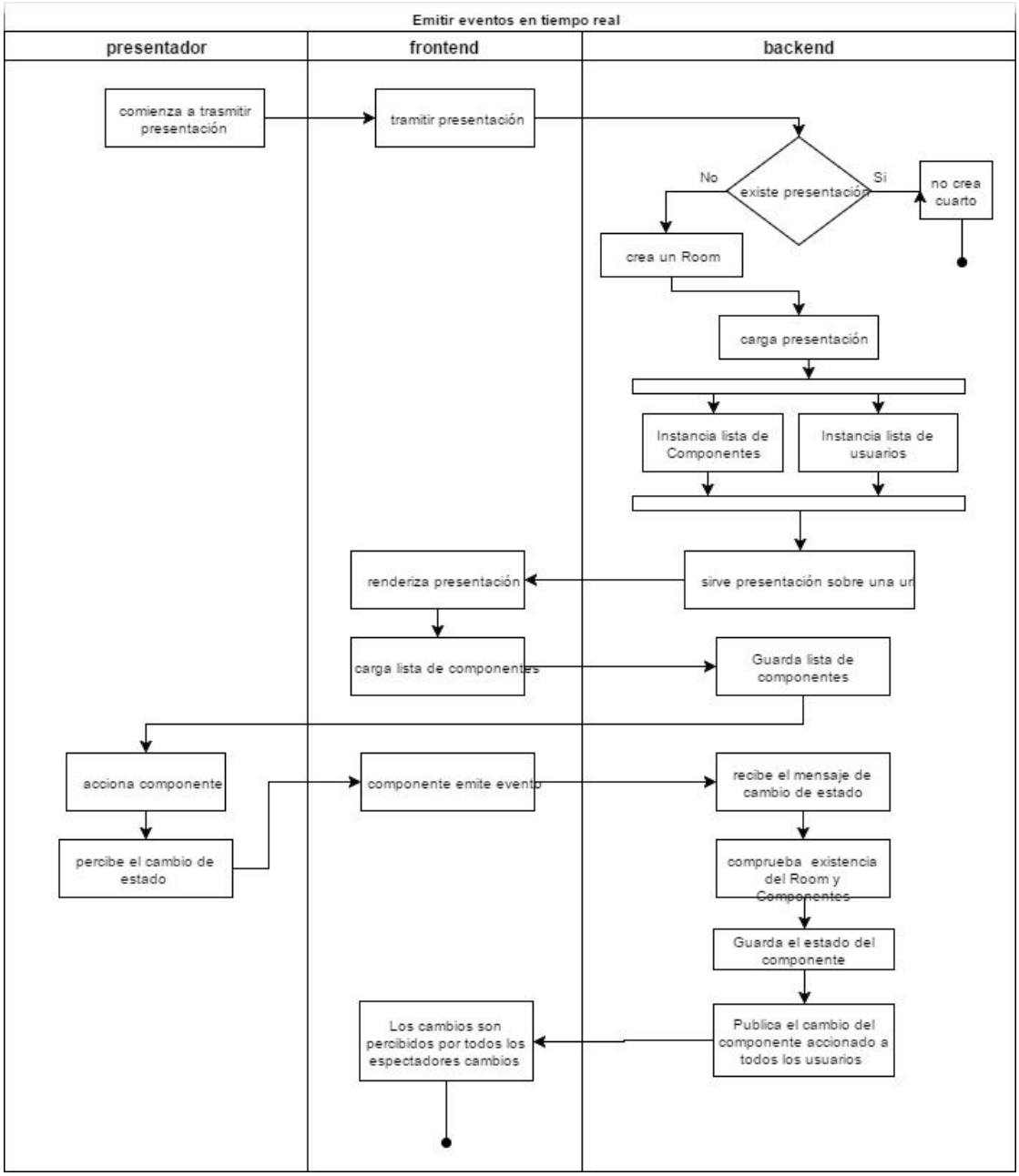
Figura 8. Diagrama de actividades - transmisión de diapositivas



5.4.3.2 Emisión de eventos en tiempo real de un componente

Este diagrama describe el flujo y el comportamiento del sistema cuando el presentador acciona un componente que emite un evento en tiempo real. Esta transmisión de eventos en tiempo real es propagado a todos los espectadores que están suscritos a la presentación publicada.

Figura 9. Diagrama de actividades - Emisión de eventos en tiempo real



5.5 QUINTA ETAPA : Evaluación y definición de Tecnología.

En esta etapa se pretende determinar qué tecnologías se aplicarán de acuerdo con los objetivos que se quieren lograr en la aplicación. Para esta evaluación se tendrá en cuenta la experiencia propia y además la orientación sobre foros, críticas, guías y documentación de la tecnologías observadas.

Debido a que esta aplicación es un prototipo web, se hace necesario definir las tecnologías que se usarán en “Frontend” o cliente que en este caso sería

la aplicación ejecutándose sobre el navegador, y las tecnologías utilizadas en el “Backend” o servidor.

En esencia este prototipo requiere mayor desarrollo sobre el Frontend, debido a que necesita tener mucha interacción con el usuario a través de una interfaz. ya que uno de los objetivos es desarrollar una estructura basada en componentes del lado del cliente y además adecuar un patrón para esta parte de la aplicación.

De igual manera para cumplir con el objetivo de emisión de eventos en tiempo real, se hace necesario contar con la participación de tecnologías que puedan mantener una comunicación sincronizada que permita tener el contenido actualizado en cada una de las diapositivas, las cuales cada uno de los usuarios esté percibiendo. Es por esto que se necesita de la comunicación con un servidor para administrar la comunicación entre el presentador y los espectadores.

En síntesis que se hace necesario definir muy bien las tecnologías utilizadas, para que cumplan con los objetivos y que además estos vayan de acuerdo con los diseños planteados.

5.5.1. Evaluación y definición del stack del lado del servidor web.

5.5.1.1 Tecnología para construir el servidor web

Desde la propia experiencia, existen muchas frameworks para crear un servidor web. Se puede encontrar distintos frameworks desarrollados en lenguajes diferentes, e incluso frameworks sobre un mismo lenguaje pero tienen características tan distintas que pueden conllevar a resolver mejor un cierto problema de arquitectura.

Por consiguiente, se hizo una evaluación entre diferentes frameworks para determinar que tecnología facilita el cumplimento de los requerimientos del sistema y es por esto que se determinó usar SailsJS[15].

SailsJS es un framework diseñado bajo el patrón de diseño MVC (Modelo Vista Controlador). Este framework como la gran mayoría que son desarrollados sobre javascript, es ejecutado sobre NodeJS[16].

[15] Página oficial SailsJS [citado:18 abril 2017] http://sailsjs.com/
[16] Michael Abernethy. ¿Simplemente que es NodeJS? [online]. IBM developersWorks, 14/06/2011-[citado:18 abril 2017]https://www.ibm.com/developerworks/ssa/opensource/library/os-nodejs/

Una de las grandes ventajas de este framework es la rapidez para montar Modelos y poder acceder a estos sin importar la base de datos, ya que provee un ORM(Object Relational Mapping) llamado Waterline. Adicionalmente se pueden generar muy ágilmente API's para insertar, modificar, leer y eliminar los datos definidos en cada uno de los modelos. También una de las ventajas de este framework, ofrece la arquitectura para soportar eventos en tiempo real bajo el protocolo de sockets. Igualmente provee una capa de seguridad basado en roles para el control de acceso a la información de la aplicación.

Es necesario recordar que NodeJS es la plataforma sobre la cual corre este framework, y esta plataforma surgió gracias al acondicionamiento del motor V8[17] que usa el navegador chrome para interpretar javascript. Por lo tanto NodeJS puede soportar eventos de entrada y salida I/O sin bloqueos, es decir que puede soportar miles de conexiones al mismo tiempo garantizando que no haya un punto de ruptura, debido a la arquitectura con la cual fue concebido, que es una arquitectura orientada a eventos. [18]

5.5.1.2 Tecnología para manejo de eventos en tiempo real.

Una vez definido que el servidor web será construido con SailsJS, se le dará utilidad a una de las características de este framework para crear toda la estructura de emisión de eventos en tiempo real.

Por esta razón cabe aclarar que SailsJS, provee esta capa gracias a **Socket.io**[19], una herramienta creada para soportar una comunicación en tiempo real de manera bidireccional, basada en eventos. Esto quiere decir que un cliente(Navegador Web) y un servidor web se puede comunicar de manera constante e inmediata si existe un cambio de estado en alguna de las dos partes. Este tipo de comunicación fue estandarizada bajo el Protocolo de Websockets realizada en 2011, el cual consiste en abrir una conversación que es permitido por trozos de mensajes a través de la capa TCP. El objetivo de esta tecnología es proveer un mecanismo para aplicaciones que estén

[17] Página oficial V8 engine, [online] [citado 17 mayo 2017] https://developers.google.com/v8/
[18] Michael Abernethy. ¿Simplemente que es NodeJS? [online]. IBM developersWorks, 14/06/2011-[citado:20 abril 2017]
https://www.ibm.com/developerworks/ssa/opensource/library/os-nodejs/
[19] Página oficial Socket.io, [online]. [citado:20 abril 2017] https://socket.io/

corriendo sobre un navegador web que necesiten dos vías de comunicación con servidores para que no tengan que abrir múltiples conexiones HTTP [20].

Por otra parte, SailsJS provee una librería para encapsultar todas las funcionalidades que provee Socket.io.

5.5.1.3 Definir Base de datos.

Actualmente existen dos tipos de bases de datos las SQL y las NoSQL. Sobre las SQL han sido bases de datos altamente usadas en los últimos tiempos, sin embargo con la aparición del procesamiento de grandes volúmenes, velocidades y tipos de datos han modificado las necesidades en los desarrolladores, por lo cual se crean las bases de datos NoSQL o no relacionales.[21]

En este aplicación es necesario aplicar una base de datos NoSQL, debido la aplicación requiere procesar datos en tiempo real, lo cual se requiere de alta velocidad de procesamientos de datos. Igualmente se deben guardar los estados de cada uno de los componentes que tiene la aplicación, y estos tienen tipos de datos tan distintos que para ser guardados en una base de datos SQL se hace muy complejo.

Atendiendo a estas consideraciones se eligió trabajar con **MongoDB**, la cual es una entre muchas bases de datos no relacionales que existen, sin embargo esta es la más usada hasta el momento. Esta base de datos es orientada a documentos y gestionan datos semi estructurados, es decir almacena los datos en algún formato estándar como XML, JSON o BSON. [22]

Para usar MongoDB con NodeJS es necesario instalar el conector entre ambas tecnologías, para esto y otros módulos más se utilizará el manejador de paquetes NPM[23]. Este es un administrador donde residen cientos de módulos desarrollados en Javascript, creados por comunidades y desarrolladores de este lenguaje. Estos paquetes en su gran mayoría tienen licencia de código abierto y son empleados para construir una aplicación de todo tipo, no solo web.

---

[20] Internet Engineering Task Force, The Websockets Protocol [online].[citado: 20 abril 2017] https://tools.ietf.org/html/rfc6455

[21] Microsoft, No SQL frente a SQL [online]. [citado:20 abril 2017] https://docs.microsoft.com/es-es/azure/documentdb/documentdb-nosql-vs-sql

[22] Rubén Fernández, Bases de datos NoSQL 27 de enero 2014.[online] [citado:20 abril 2017] https://www.genbetadev.com/bases-de-datos/bases-de-datos-nosql-elige-la-opcion-que-mejor-se-adapte-a-tus-necesidades

[23] Página oficial NPM, [online]. [citado:20 abril 2017] https://www.npmjs.com/

5.5.2. Evaluación y definición stack utilizado sobre el lado del cliente

Existen todo tipo de librerías, frameworks para crear aplicaciones web del lado del cliente bajo el lenguaje javascript, el cual está de manera predeterminada para construir aplicaciones en todos los navegadores web actualmente. Es por esto que se escogerá tecnologías que apoyen la arquitectura que se quiere lograr basada en componentes.

5.5.2.1. Tecnología utilizada para la construcción de componentes.

Para esta aplicación se determinó utilizar AngularJS en su versión 1.6.4, que es un framework MVC (Modelo Vista Controlador), diseñado para desarrollar aplicaciones web tipo SPA (Single-Page Applications), las cuales hacen el enrutamiento sobre una misma página, sin necesidad hacer recargas o peticiones al servidor una una vista, todo el funcionamiento esta sobre el cliente. Sin embargo no se utilizará está parte de enroutamiento, ya que se contará con las vistas del lado del servidor.

Uno de los mayores motivos que conllevo a utilizar este framwork, fue el hecho que este permite crear componentes/directivas las cuales se encargan de dividir el funcionamiento de la aplicación y así cumplir con los requerimientos del sistema.

De igual manera AngularJS se caracteriza por la estandarización de codificación dentro de la aplicación, lo cual lo convierte en un framework bastante rígido en su estructura, convirtiendolo en una ventaja en el momento de inclusión a mienbros del equipo, los cuales desde inicio se rigen a estos estándares.

También cabe denotar que existe una versión actual de Angular que es la 4.0 [24], se tuvo en cuenta para el desarrollo de la aplicación, pero esta versión actualmente se encuentra en fase desarrollo y su tamaño es grande y contiene varias dependencias, lo cual volve más complejo para unn usuario la creación de una diapositva de manera local.

5.5.2.2. Librerías y plugins

Para el uso de librerias y plugins, se empleara bowerJS como manejador de paquetes del lado del cliente. lo cual se instalarán las siguientes librerias:

- Bootstrap v 4.0.0-alpha: Utilizado en la maquetación y estilos básicos en plataforma administrativa de diapositivas.

[24] Página oficial Angular 4.0 , [online] . [citada 17 de Mayo de 2017] https://angular.io/

- Fontawesome 4.3.0 : Iconos como fuente de letra, utilizado en el panel administrativo
- Jquery 2.1.4 : empleado para acceder de una manera más rápida al DOM(Document Object Model) utilizado junto con AngularJS
- Simple-line-icons : Iconos como fuente de letra, utilizado sobre la parte administrativa de las diapositivas.

5.6. SEXTA ETAPA: Desarrollo de la aplicación

Sobre esta etapa del proyecto se pretende hacer una descripción sobre el desarrollo de la aplicación, para ello se tiene en cuenta dos partes fundamentales en la aplicación.

5.6.1 Desarrollo del lado del servidor

Ya que el servidor web fue construido con SailsJS, se hace necesario revisar su documentación http://sailsjs.com/get-started para comenzar a configurarlo. Previamente tener instalado NodeJS y NPM y seguir las instrucciones sobre la documentación.

Una vez generado el servidor web, procede a configurar las diferentes rutas y plantillas html las cuales se van a montar las diferentes vistas de Login, Registro y Panel administrativo. Para cada vista existe una controlador y un Modelo. Este último es generado fácilmente por medio de comandos gracias a que SailsJS cuenta con un ORM (Object Relacional Mapping) para crearlos sin necesidad tener definido una base de datos.

Sabiendo que MongoDB es la base de datos empleada para este proyecto, se instala sobre la máquina y se configura sobre un archivo de conexión de la base de datos con el ORM de Sails, llamado connections.js.

En esta parte se crear políticas de acceso a las diferentes vistas e información de la aplicación, esto también hace parte de una configuración que tiene SailJS llamado policies.

En cuanto al soporte de eventos de tiempo real, se hace necesario utilizar un módulo nativo de SailsJS de websockets construido bajo el framework Socket.io. Este Módulo abstrae las funcionalidades para construir la aplicación bajo el patrón publisher/subscriber , el cual es utilizado para transmitir las presentación en tiempo real junto con los componentes internos.

5.6.2 Desarrollo del lado del cliente

Con el motivo de agilizar el proceso de desarrollo en esta aplicación sobre la maquetación HTML y CSS, se implementó Modular Admin[25], el cual un proyecto de código abierto que determina un diseño de plantillas de un panel administrativo. Por lo tanto con este diseño de plantillas se adaptan las diferente vistas.

Por otra parte para tener vistas dinámicas y reactivas a cambios en tiempo real, se hace uso del framework AngularJS, el cual provee un conjunto de patrones y técnicas para estandarizar el desarrollo de aplicaciones web con javascript sobre el navegador.

Es por lo tanto que con AngularJS se apoya en su totalidad para construir este prototipo web. En definitiva fue desarrollado dos módulos básicos que se conectan entre sí para la administración de diapositivas, y otro módulo para la presentación y el uso de componente que emiten los eventos en tiempo real.

Con el módulo de diapositivas fue desarrollado para ser utilizado de manera abstraída de la plataforma de diapositivas, es decir se puede utilizar independientemente para la creación de diapositivas como si fuera un framework. Lo que permite desarrollar toda la diapositiva de manera local, para luego copiar un contenido básico dentro de la edición de diapositivas en la plataforma.

En lo que respecta a la construcción de diapositivas se utilizó el layout o distribución de RevealJS, y se incluyó los términos de derechos de autor para reproducir parcialmente este framework.

[25] Modular Admin, Repositorio en github [online] [consultado 17 de Mayo de 2017] ttps://github.com/modularcode/modular-admin-html

6. CONCLUSIONES

- Según los planteamientos sobre el diseño poco escalable del framework RevealJS, se pudieron evidenciar en el momento de hacer una revisión del código, por lo cual conlleva a realizar un propuesta de diseño diferente para obtener una mayor mantenibilidad.

- Desarrollar una aplicación web del lado del cliente basada en componentes hace mucho más ágil, mantenible, reusable, escalable la arquitectura, ya que esta es una de las filosofías de este tipo de diseños.

- Contar desde un inicio con estilos arquitectónicos, patrones de diseño y arquitectónico, en una etapa temprana de diseño de la aplicación previene tener problemas con una arquitectura en el momento de implementación. Esto se debe a que existe una planificación para el diseño preventivo y no correctivo.

- Utilizar el patrón de diseño publisher/subscriber para la emisión de eventos en tiempo real, permite agregar el bajo acoplamiento entre los diferentes componentes cuando emiten y reciben una acción. Adicionalmente hace escalar la cantidad de componentes suscritos a un tema/acción determinada por los mismos, y a su vez los usuarios espectadores pueden seguir percibiendo estos cambios en tiempo real sin problemas.

7. RECOMENDACIONES

- Para desarrollar una arquitectura se recomienda, que las partes implicadas conozcan bien los objetivos a desarrollar y entre estos los atributos calidad los cuales se quieren lograr.

- En el momento de agregar una estructura para soportar eventos en tiempo real del lado del cliente, la mejor manera es agregar una interfaz para detectar eventos y no incluirlos dentro de cada componente que deba emitir eventos en tiempo real, para manejar bajo acoplamiento entre acciones y componentes.

- En efectos de determinar el tipo de evaluación realizada a una arquitectura, es importante tener en cuenta en qué momento se aplicará dicha evaluación, ya que no todas las evaluación desenvuelven bajo los mismo objetivos y parámetros.

- Se recomienda utilizar una pila/stack de tecnologías de acuerdo a los objetivos y atributos de calidad escogidos. Adicionalmente que estas tengan apoyo documental sobre su uso, para evitar desconocimientos de usos en la etapa de desarrollo.

- Se recomienda que en posteriores versiones de la aplicación se desarrolle sobre la tecnología Web Components, la cual provee una mejor sintaxis y semántica en el desarrollo.

BIBLIOGRAFÍA.

Buitrago Ortega Cristian. Componentes de software. [online].[citado 08 de Abril de 2012]
http://qualityandprogramming.blogspot.com.co/2012/04/que-es-un-componente-de-softwarepara-el.html

Casal Terreros Julio. Desarrollo de software basado en componentes. [online]. [citado 21 de Marzo de 2011]
https://msdn.microsoft.com/es-es/library/bb972268.aspx 21/03/2011,

Erika Camacho, Fabio Cardeso, Gabriel Nuñez. Guía Arquitectura de Software [documento digital].

Mateusz Papiernik. How install MongoDB on Ubuntu 16.04.[publicado: 5 de mayo de 2016] [online]. [citado 21 de Mayo de 2017]
https://www.digitalocean.com/community/tutorials/how-to-install-mongodb-on-ubuntu-16-04.

Mike McNeil, Irl Nathan.Sails.js in Action.[Enero de 2017] [ebook].

Mike McNeil, Irl Nathan. Guía de Referencia SailsJS. [online]. [citado 21 de Mayo de 2017] http://sailsjs.com/documentation/reference

Jimmy Collazos, AngularJS Directives [video tutorial]. Lista publicado 23 de abril de 2014
https://www.youtube.com/playlist?list=PLgEQ01FK0MFaCoCvRhlw3VP5Tw9EhmWeq

Gevorg Harutyunyan. Modular admin [repositorio]. Repositorio en github publicado 19 de Julio de 2015.
https://github.com/modularcode/modular-admin-html

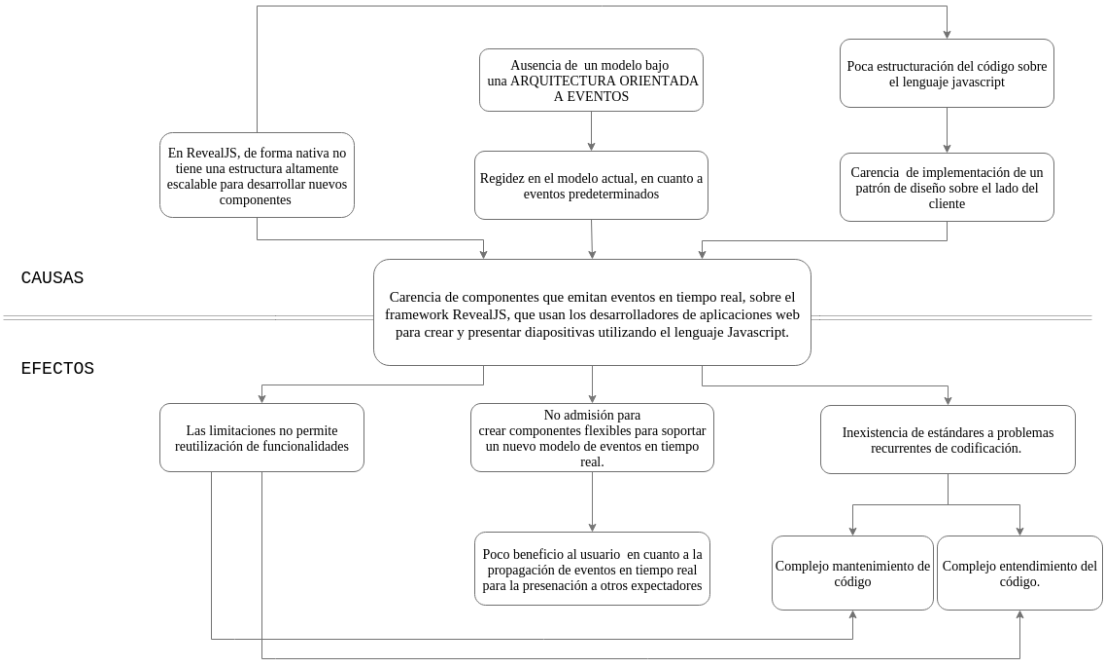
Oscar Gallón. Tutoriales SailsJS [video tutorial]. Lista de video tutoriales, subido a youtube.[20 de Octubre de 2015]
https://www.youtube.com/playlist?list=PLefWQQrJ1fC6PbyNj5Ckl57nrgE-GLR2H

Andrea Delgado, Alberto Castro, Martín Germán. Evaluación de Arquitecturas de Software con ATAM (Architecture Tradeoff Analysis Method): un caso de

estudio [online]. [Ed: 2004] Universidad de la República, Facultad de Ingeniería, Instituto de Computación [2005], Revisado en el 2005.

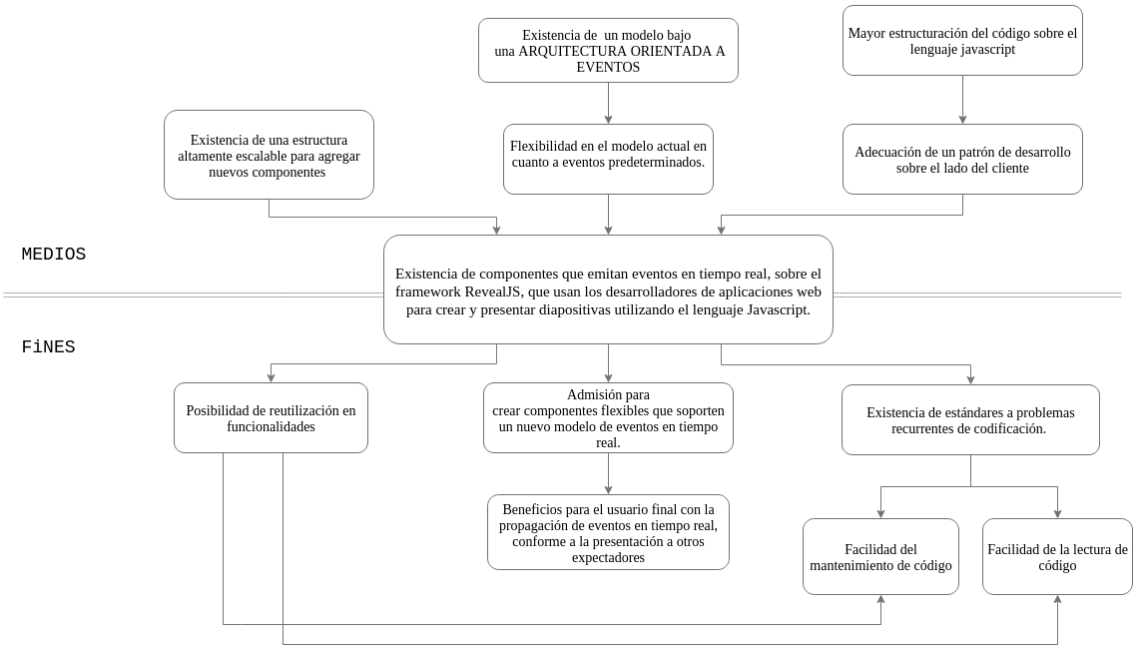
LISTA DE ANEXOS

ÁRBOL DE PROBLEMAS



Anexo B. Árbol de objetivos

ÁRBOL DE OBJETIVOS



Anexo C. Repositorio del proyecto en GitHub

El repositorio se encuentra abierto a cualquier persona interesada en contribuir. La documentación de la aplicación se encuentra dentro del archivo README.md del respositorio.

https://github.com/jdmorales/slides_remote